Introduction to Statistics Using R
(Statistical Analysis of Genome Sequences)

Yuzhen Ye (yye@indiana.edu)
School of Informatics and Computing, IUB
Given a sequence, what can we ask?

AGCTTTTCATTTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTTAAAAAAGAGTGTCTGATAGCAGC
TTCTGAACTGTTTACCTGGCCTGGATTTAAAAATTTTTATTGACTTAGTCACTAAATACCTTTAACC
TATAGGCATAGCGACAGACAGATAAAAATTACAGAGTACACAACATCCATGAAACGCATTAGCACCACC
ATTACCACACATTACCACTCATTACCCAGGTAAACGGGTCCGGCTGACGGGTACAAGGAAACACAGAA
CCGACCTGACAGTGCGGGCTTTTTTTTTCGACCAAGGTAACGAGGTAACCAACATGCAGTGTGGAA
GTTCGGCGGTCATCATGTCAGGGAAATGGCAGAAGCTTTTCTGCGTTGGCGATATTCTGGAAGCAATGCC
AGGCAGGGGCAGGTGGCCACCCCCTCTCTGCCCCCCCCAATAATCAACCCACACCTGGCTGGATATGG
AAAAAACCATTAGCGCCAGGATGCTTTTACCCAAATATCAGCGATGCAGGAACGTATTTTCTGCCGAACTTT
GACGCGACTCAGCCGCCGCCAGGGGTCCCGCTGGCGCAATTGAAATTTTTCTGCGATCAGGAAATTT
GCCCAAAATAAACATGTCCTGGCATGGCATTTGTTTGGCGCTGCACTGGCATAGCATCAACGCTGGCGC
TGATTGGCGGTGGCGAGAAAATGTCGATCGCCATTATGGCCGGCGTATTAGAAGCGCGGCGGTCAACAAGGT

• **What sort of statistics should be used to describe this sequence?**
• **Can we determine what sort of organism this sequence came from based on sequence content?**
• **Do parameters describing this sequence differ (significantly) from those describing bulk DNA in that organism?**
• **What sort of sequence might this be: protein coding? Transposable elements?**

**Ref:** Computational Genome Analysis, Deonier et al.
Why we need statistics

- Everything varies
  - If you measure the same thing twice you will get two different answers
  - Heterogeneity is universal: spatial heterogeneity & temporal heterogeneity
  - We need to distinguish between variation that is scientifically interesting, and variation that just reflects background heterogeneity

- Significance ("statistically significant")
  - A result is unlikely to have occurred by chance
  - A result is unlikely to have occurred by chance if the null hypothesis was true
  - Null hypothesis says that “nothing’s happening”, and the alternative says “something is happening”; null hypothesis has to be a falsifiable hypothesis
P values

- A $p$ value is an estimate of the probability that a particular result, or a result more extreme than the result observed, could have occurred by chance, if the null hypothesis were true.
- The $p$ value is a measure of the credibility of the null hypothesis.
- For example, when comparing two sample means, a small $p$ value means that the null hypothesis (the two means are the same) is unlikely to be true and the difference is statistically significant.
Why R?

- R first appeared in 1996, when the statistics professors Robert Gentleman, left, and Ross Ihaka released the code as a free software package.
- “Google, for example, taps R for help understanding trends in ad pricing and for illuminating patterns in the search data it collects. Pfizer has created customized packages for R to let its scientists manipulate their own data during nonclinical drug studies rather than send the information off to a statistician.”
## Comparison of data analysis packages

<table>
<thead>
<tr>
<th>Name</th>
<th>Advantages</th>
<th>Disadvantages</th>
<th>Open source?</th>
<th>Typical users</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Library support; visualization</td>
<td>Steep learning curve</td>
<td>Yes</td>
<td>Finance; Statistics</td>
</tr>
<tr>
<td>Matlab</td>
<td>Elegant matrix support; visualization</td>
<td>Expensive; incomplete statistics support</td>
<td>No</td>
<td>Engineering</td>
</tr>
<tr>
<td>SciPy/NumPy/</td>
<td>Python (general-purpose programming</td>
<td>Immature</td>
<td>Yes</td>
<td>Engineering</td>
</tr>
<tr>
<td>Matplotlib</td>
<td>language)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Excel</td>
<td>Easy; visual; flexible</td>
<td>Large datasets</td>
<td>No</td>
<td>Business</td>
</tr>
<tr>
<td>SAS</td>
<td>Large datasets</td>
<td>Expensive; outdated programming language</td>
<td>No</td>
<td>Business; Government</td>
</tr>
<tr>
<td>Stata</td>
<td>Easy statistical analysis</td>
<td></td>
<td>No</td>
<td>Science</td>
</tr>
<tr>
<td>SPSS</td>
<td>Like Stata but more expensive and worse</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Objectives

- R basics
  - Data frame, lists, matrices
  - I/O (read.table)
  - Graphical procedures
- How to apply R for statistical problem?
- How to program your R function?
- Statistics basics

- R website: [http://www.r-project.org/](http://www.r-project.org/) (check out its documentation!)
Working with R

- Most packages deal with statistics and data analysis.
- You can run R on different platforms
- Knowing where you are
  - `getwd()` Get Working Directory
  - `setwd()` Set Working Directory
  - `list.files()` List the Files in a Directory/Folder
- Getting quick help with `help()`, `demo()`, `example()`
  - `help(plot)`
  - `demo(nlm)` #Nonlinear least-squares
  - `example()` #example("smooth", package="stats",
    lib.loc=.Library)"
Packages in R environment

- Basic packages
  - "package:methods"  "package:stats"
    "package:graphics"  "package:utils"
    "package:base"

- Contributed packages

- Bioconductor
  - an open source and open development software project for the analysis and comprehension of genomic data

- You can see what packages loaded now by the command `search()`

- Install a new package?
  - `install.packages("Rcmdr", dependencies=TRUE)"`
R basic data types

- **vector**, **array**, **list**, **matrix**, **data frame**
  - **list**: an ordered collection of data of *arbitrary types*.
  - **vector**: an ordered collection of data of the same type.
  - **matrix**: all elements of a matrix have the same mode, i.e. all numeric, or all character. Thus a matrix is a more restricted structure than a data frame.
  - **array**: The generalization from a matrix (2 dimensions) to allow > 2 dimensions gives an array. A matrix is a 2D array.
  - **data frame**: A data frame is a generalization of a matrix, in which different columns may have different modes. All elements of any column must however have the same mode, i.e. all numeric or all factor, or all character.
A data frame is an object with rows and columns (a bit like a 2D matrix)
  - The rows contain different observations from your study, or measurements from your experiment
  - The columns contain the values of different variables
All the values of the same variable must go in the same column!
  - If you had an experiment with three treatments (control, pre-heated and pre-chilled), and four measurements per treatment
### What’s the correct data frame?

<table>
<thead>
<tr>
<th>Control</th>
<th>Pre-heated</th>
<th>Pre-chilled</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>6.3</td>
<td>7.1</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
<td>..</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Response</th>
<th>Treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>Control</td>
</tr>
<tr>
<td>5.9</td>
<td>Control</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>..</td>
<td>..</td>
</tr>
<tr>
<td>6.2</td>
<td>Pre-heated</td>
</tr>
</tbody>
</table>
Factors

- Factors: classification variables

```r
> trt <- factor(rep(c("Control", "Treated"), c(3, 4)))
> str(trt)
  Factor w/ 2 levels "Control","Treated": 1 1 1 2 2 2 2
> summary(trt)  # summary gives a frequency table
  Control Treated
     3        4
```

- If the levels of a factor are numeric (e.g. the treatments are labelled “1”, “2”, and “3”) it is important to ensure that the data are actually stored as a factor and not as numeric data. Always check this by using `summary`. 
Assigning values to variables

- R uses ‘gets’ <- rather than the more familiar ‘equals’ = sign
  - x <- 12.6  #assign value to a numerical variable
  - y <- c(3, 7, 9, 11) #vector
  - a <- 1:6 # : means a series of integers between
  - b <- seq(0.5, 0, -0.1)
Data input from a file

Learning how to read the data into R is amongst the most important topics you need to master

From the file

– `read.table()`
Obtain parts of your data

- **Subscripts with vectors**
  
  - `y[3]`  #third element
  - `y[3:7]`  #from third to 7th elements
  - `y[c(3, 5, 6, 9)]`  #3rd, 5th, 6th, and 9th elements
  - `y[-1]`  #drop the first element; dropping using negative integers
  - `y[y > 6]` #all the elements that are > 6

- **Subscripts with matrices, arrays, and dataframe**
  
  - `A <- array(1:30, c(5, 3, 2))` #3D array
    - `A[,2:3,]`
    - `A[2:4,2:3,]`
  - `worms <- read.table("worms.txt", header=T, row.names=1)`
    - `worms[,1:3]` #all the rows, columns 1 to 3

- **Subscripts with lists**
  
  - `cars <- list(c("Toyota", "Nissan"), c(1500, 1750), c("blue", "red", "black"))`
    - `cars[[3]]`  # c("blue", "red", "black")
    - `cars[[3]][2]`  # "red"  note: not cars[3][2]`
Using logic conditions to get subsets

```r
> library(lattice)
> barley[1:7,]
          yield  variety year            site
1 27.000000 Manchuria 1931 University Farm
2 48.866667 Manchuria 1931          Waseca
3 27.433334 Manchuria 1931          Morris
4 39.933333 Manchuria 1931       Crookston
5 32.966667 Manchuria 1931    Grand Rapids
6 28.966667 Manchuria 1931    Grand Rapids
7 43.066666 Glabron 1931 University Farm

> Duluth1932 <- barley[barley$year=="1932" & barley$site=="Duluth", c("variety","yield")]
```
Save object/data

- Every R object can be stored into and restored from a file with the commands “save” and “load”.
  - > save(x, file="x.Rdata")
  - > load("x.Rdata")

- Importing and exporting data with rectangular tables in the form of tab-delimited text files.
  - > write.table(x, file="x.txt", sep="\t")
Write R function

- A function definition looks like

```r
funcdemo <- function(x, y)
{
    z <- x + y
    return (z)
}
```
Control flow

- if(cond) expr
- if(cond) cons.expr else alt.expr
- for(var in seq) expr
  - for (i in 1:n)
- while(cond) expr
- repeat expr
- break & next

```r
for(i in 1:10) {
  print(i*i)
}
```

```r
i<-1
while(i<=10) {
  print(i*i)
  i<-i+sqrt(i)
}
```
<table>
<thead>
<tr>
<th>Statistical functions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rnorm, dnorm, pnorm, qnorm</td>
<td>Normal distribution random sample, density, cdf and quantiles</td>
</tr>
<tr>
<td>lm, glm, anova</td>
<td>Model fitting</td>
</tr>
<tr>
<td>loess, lowess</td>
<td>Smooth curve fitting</td>
</tr>
<tr>
<td>sample</td>
<td>Resampling (bootstrap, permutation)</td>
</tr>
<tr>
<td>.Random.seed</td>
<td>Random number generation</td>
</tr>
<tr>
<td>mean, median</td>
<td>Location statistics</td>
</tr>
<tr>
<td>var, cor, cov, mad, range</td>
<td>Scale statistics</td>
</tr>
<tr>
<td>svd, qr, chol, eigen</td>
<td>Linear algebra</td>
</tr>
</tbody>
</table>
Graphical procedures in R

- High-level plotting functions create a new plot on the graphics device, possibly with axes, labels, titles and so on.
- Low-level plotting functions add more information to an existing plot, such as extra points, lines and labels.
- Interactive graphics functions allow you interactively add information to, or extract information from, an existing plot, using a pointing device such as a mouse.
Knowing your data

- Types of your data
- Central tendency
  - Mode: The data values that occur most frequently are called the **mode** (drawing a histogram of the data)
  - Arithmetic mean: \( \bar{a} = \frac{\sum a}{n} \)
    - \( \text{sum}(a) / \text{length}(a) \)
  - Median: the “middle value” in the data set
    - \( \text{sort}(y)[\text{ceiling}(\text{length}(y)/2)] \)
- Variance
  - Degrees of freedom (d.f.)
Summary of your data

- Commands
  - `summary()`
  - `mean()`
  - `var()`, `sd()`
  - `min()`, `max()`
  - `hist()`
  - `boxplot()`
## Statistical tests

<table>
<thead>
<tr>
<th>Data type</th>
<th>Goal</th>
<th>Measurement (from Gaussian Population)</th>
<th>Rank, Score, or Measurement (from Non-Gaussian Population)</th>
<th>Binomial (Two Possible Outcomes)</th>
<th>Survival time</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Describe one group</td>
<td></td>
<td>Mean, SD</td>
<td>Median, interquartile range</td>
<td>Proportion</td>
<td>Kaplan Meier survival curve</td>
</tr>
<tr>
<td>Compare one group to a hypothetical value</td>
<td></td>
<td>One-sample t test</td>
<td>Wilcoxon test</td>
<td>Chi-square or Binomial test</td>
<td></td>
</tr>
<tr>
<td>Compare two unpaired groups</td>
<td></td>
<td>Unpaired t test</td>
<td>Mann-Whitney test</td>
<td>Fisher's test (chi-square for large samples)</td>
<td>Log-rank test or Mantel-Haenszel</td>
</tr>
<tr>
<td>Compare two paired groups</td>
<td></td>
<td>Paired t test</td>
<td>Wilcoxon test</td>
<td>McNemar's test</td>
<td>Conditional proportional hazards regression</td>
</tr>
</tbody>
</table>

Parametric/Nonparametric tests

- Choose a parametric test if you are sure that your data are sampled from a population that follows a Gaussian distribution (at least approximately) (e.g., t test, Fisher test).

- You should definitely select a nonparametric test in three situations:
  - The outcome is a rank or a score and the population is clearly not Gaussian
  - Some values are "off the scale," that is, too high or too low to measure
  - The data are measurements, and you are sure that the population is not distributed in a Gaussian manner

- Large data sets present no problems: The central limit theorem ensures that parametric tests work well with large samples even if the population is non-Gaussian.
Statistical modeling

- It is not “the data is fitted to a model”; rather, it is “the model is fitted to the data”
- To determine a minimal adequate model from the large set of potential models that might be used to describe the given set of data
- We define the “best” model in terms of maximum likelihood
  - Given the data
  - And given our choice of model
  - What values of the parameters of that model make the observed data most likely?
## Probability distribution

<table>
<thead>
<tr>
<th>law</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian (normal)</td>
<td>rnorm(n, mean=0, sd=1)</td>
</tr>
<tr>
<td>exponential</td>
<td>rexp(n, rate=1)</td>
</tr>
<tr>
<td>gamma</td>
<td>rgamma(n, shape, scale=1)</td>
</tr>
<tr>
<td>Poisson</td>
<td>rpois(n, lambda)</td>
</tr>
<tr>
<td>Weibull</td>
<td>rweibull(n, shape, scale=1)</td>
</tr>
<tr>
<td>Cauchy</td>
<td>rcauchy(n, location=0, scale=1)</td>
</tr>
<tr>
<td>beta</td>
<td>rbeta(n, shape1, shape2)</td>
</tr>
<tr>
<td>‘Student’ (t)</td>
<td>rt(n, df)</td>
</tr>
<tr>
<td>Fisher–Snedecor (F)</td>
<td>rf(n, df1, df2)</td>
</tr>
<tr>
<td>Pearson (χ²)</td>
<td>rchisq(n, df)</td>
</tr>
<tr>
<td>binomial</td>
<td>rbinom(n, size, prob)</td>
</tr>
<tr>
<td>geometric</td>
<td>rgeom(n, prob)</td>
</tr>
<tr>
<td>hypergeometric</td>
<td>rhyper(nn, m, n, k)</td>
</tr>
<tr>
<td>logistic</td>
<td>rlogis(n, location=0, scale=1)</td>
</tr>
<tr>
<td>lognormal</td>
<td>rlnorm(n, meanlog=0, sdlog=1)</td>
</tr>
<tr>
<td>negative binomial</td>
<td>rnbinom(n, size, prob)</td>
</tr>
<tr>
<td>uniform</td>
<td>runif(n, min=0, max=1)</td>
</tr>
<tr>
<td>Wilcoxon’s statistics</td>
<td>r wilcox(nn, m, n), r signrank(nn, n)</td>
</tr>
</tbody>
</table>
(Generalized) linear models

- The model formulae look very like equations but there are important differences
  \[ y = a + bx \quad (\text{formula: } y \sim x) \]
  \[ y = a + bx + cz \quad (\text{formula: } y \sim x + z) \]

- Fitting linear models
  \[
  \text{fm2} \leftarrow \text{lm}(y \sim x1 + x2, \text{data} = \text{production})
  \]

- Generalized Linear Models
  \[
  \text{glm}(y \sim z, \text{family} = \text{poisson})
  \]
  \[
  \text{glm}(y \sim z, \text{family} = \text{binomial})
  \]
Microarray data analysis

Pre-processing

Differential expression
- sngenes
- genefilter
- limma
- multtest

Graphs & networks
- graph
- RBGL
- Rgraphviz

Cluster analysis
- CRAN
  - class
  - cluster
  - MASS
  - mva

Prediction
- CRAN
  - class
  - e1071
  - ipred
  - LogitBoost
  - MASS
  - nnet
  - randomForest
  - rpart

Annotation
- annotate
- annaffy
- metadata
- packages

Graphics
- geneplotter
- hexbin
- + CRAN
### Useful R/BioConductor packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marray, limma</td>
<td>Spotted cDNA array analysis</td>
</tr>
<tr>
<td>affy</td>
<td>Affymetrix array analysis</td>
</tr>
<tr>
<td>vsn</td>
<td>Variance stabilization</td>
</tr>
<tr>
<td>annotate</td>
<td>Link microarray data to metadata on the web</td>
</tr>
<tr>
<td>ctest</td>
<td>Statistical tests</td>
</tr>
<tr>
<td>genefilter, limma,</td>
<td>Gene filtering (e.g.: differential expression)</td>
</tr>
<tr>
<td>multtest, siggenes</td>
<td></td>
</tr>
<tr>
<td>mva, cluster, clust</td>
<td>Clustering</td>
</tr>
<tr>
<td>class, rpart, nnet</td>
<td>Classification</td>
</tr>
</tbody>
</table>
Example 1: Primate’s body weight & brain volume

<table>
<thead>
<tr>
<th>Species</th>
<th>Bodyweight</th>
<th>Brain Vol</th>
</tr>
</thead>
<tbody>
<tr>
<td>H. sapiens</td>
<td>54000</td>
<td>1350</td>
</tr>
<tr>
<td>H. erectus</td>
<td>55000</td>
<td>804</td>
</tr>
<tr>
<td>H. habilis</td>
<td>42000</td>
<td>597</td>
</tr>
<tr>
<td>A. robustus</td>
<td>36000</td>
<td>502</td>
</tr>
<tr>
<td>A. afarensis</td>
<td>37000</td>
<td>384</td>
</tr>
</tbody>
</table>

- Summary of the data (bodyweight, and brainvol)
- Correlation between bodyweight and brainvol
- Linear fitting
- Plotting
Histogram of bodyweight

Histogram of brainvol

Brain volume versus body weight
Example 2: Gene length

- Do the protein-coding genes in *E.coli* and ? Genomes have statistically different gene lengths?
Example 3: word distribution analysis for genome sequences

- HW2

- Sequence alignment next week!